ECE 573
Problem Set 6: Instruction Scheduling

For the following problems, assume we are given an architecture and instruction set with the following specifications:

Functional Units

- One arithmetic logic unit, called ALU
- One multiply/add unit (not pipelined), called MAU
- One memory unit, called MEM

Instructions

- **ADD**: This instruction has a one cycle latency, and can execute on either the ALU or the MAU
- **MUL**: This instruction has a two cycle latency, and can execute on the MAU
- **LD** or **ST**: These instructions occupy either the ALU or the MAU for one cycle, and then the MEM in the next cycle.

Code For these problems, consider the following piece of code, which implements $w = x * y + z$:

```
1: LD X, T1
2: LD Y, T2
3: MUL T1, T2, T3
4: LD Z, T4
5: ADD T3, T4, T5
6: ST T5, W
```

Problems

1. Give the reservation tables for each of the instructions supported by this architecture.
2. Draw the resource constraint finite automaton for this architecture
3. Draw the data dependence graph for the program, including latency labels (you may omit distance labels)

4. Assume that we are using height-based priorities for list scheduling. Give the schedule for this program. For each cycle, list the three functional units. If an instruction is newly scheduled in that cycle, write the instruction number in the entry for the functional unit it is occupying. If a functional unit is still occupied from a previously scheduled instruction, put an "x" in that slot.

5. Give another legal schedule for this program which takes longer to run than the one you produced in the previous question (without inserting gratuitous no-ops).